重复数据删除技术的发展综述

王树鹏 云晓春 郭莉

摘要：随着信息化程度的不断提高，数据量不断爆炸式增长。这给数据存储管理带来了越来越大的压力，特别是数据中所存在的重复内容给存储空间造成了极大的浪费。为了提高存储空间利用率，重复数据删除（数据消冗）技术应运而生。本文就重复数据删除技术的背景、分类、关键技术、应用场景、发展现状及未来发展趋势进行了总结分析。

关键词：重复数据删除 数据消冗 存储优化

1 技术背景

数据量的爆炸式增长以及大规模集中使得数据重复所导致的空间浪费问题越来越严重，这促使了数据消冗技术的出现和发展。目前，数据消冗方面的研究工作已经在消冗率提升、性能优化以及可靠性保证等方面取得了一系列有价值的成果，有效地推动了该技术的应用。在面向服务的云存储系统中，用户数据的大规模集中使得数据消冗更加必要，同时也对该技术提出了新的挑战。为了满足不同用户对消冗率、可靠性以及性能方面的不同需求，存储系统必须能够对包括消冗率、可靠性及性能的服务质量进行有效的控制和调节，同时存储系统规模的增加也使得节能、保证服务质量（QoS）以及降低能耗成为数据消冗存储中需要解决的关键问题。目前数据消冗的研究成果都无法完全解决这些问题，不能满足面向服务的大规模存储系统的实际需要。作者所申请的自然科学基金项目“面向服务质量和能耗优化的数据消冗存储技术研究”将重点针对该问题进行研究，目标是建立面向服务质量和能耗优化的数据消冗理论和技术体系。我们首先研究了数据消冗的关键属性，提出了数据消冗服务质量描述和评价方法体系；然后对数据消冗的能耗进行了研究，提出了面向数据消冗的能耗分析方法及能耗优化策略；最后，研究了多目标数据消冗技术，基于服务质量收益和能耗代价函数，实现了面向服务质量和能耗优化的数据消冗技术。本课题的研究内容符合存储技术的发展需求，具有重要的理论意义和实际价值。

下面对重复数据删除技术的相关背景及发展状况进行分析和阐述。

重复数据删除(数据消冗)的目的是从全局上消除存储系统中存在的冗余数据，包括文件内部以及文件之间的冗余数据，而传统的数据压缩只能够消除文件内部的冗余信息。相比之下，重复数据删除技术的数据压缩效果更加明显，针对具体应用数据的重复数据删除率可达 300:1 甚至更高，而数据压缩技术的消冗率只有 2:1 左右。因为重复数据删除技术对数据存储具有重大的作用和意义，近年来该技术也得到了广泛的关注，成为数据存储领域的研究热点[1,3,4,5,6,7,8]，关注度在数据存储和保护领域的 5 个研究热点问题中处于第一位。

在初期，大量的研究工作都集中在重复数据删除率的提升方面，通过不断减小重复数据删除粒度来提高重复数据删除率。EMC 的 Centera 系统[13]、windows 的单实例存储系统[14]采用了以文件为单位的重复数据删除方法。该方法的优点是实现简单、计算速度快，但是检测粒度较粗，重复数据删除效果差。为了提高重复数据删除率，研究者提出了定长块的检测方法，将一个文件分成固定长度的数据块，以数据块为单位进行消冗。该方法的优点是计算速度快、对数据变化反应比较敏感，主要缺陷是在文件中部分内容被插入或者修改的情况会严重影响重复数据删除效果。该方法被应用到了 Venti 归档存储系统中[5] 。为了进一步提高

消冗率，研究者针对定长块重复数据删除存在的问题，提出了变长块的重复数据删除方法，即使用拉宾指纹（Rabin fingerprint）技术或者其它相似函数确定数据块的边界[15]，将变化的内容划分到一个数据块中，典型的应用有 Shark[8]、Deep Store[7]等。另外，研究者还提出了字节级的重复数据删除机制[11]，先查找相似度比较高的数据块，然后采用差异压缩机制计算数据块之间的差异，仅存储差异部分的内容，常用的差异压缩算法有 zDelta 等[12]。

随着重复数据删除技术在海量存储系统中的应用，重复数据删除机制对存储系统吞吐率的影响逐渐体现出来，重复数据删除的性能问题逐渐引起了研究者的关注。文献[4,16,20,24]针对该问题展开了一系列的研究：文献[4]提出通过布隆过滤（Bloom Filter）、基于局部性的缓存等机制来降低消冗过程中的磁盘读写次数，提高数据重复数据删除速度；文献[16]通过将数据块打包成定长的数据对象来提高数据读写性能；文献[20]提出了一个两阶段重复数据删除机制，通过将随机的小的磁盘读写调整为序列化的大的磁盘读写来提高重复数据删除的吞吐率；文献[24]利用局部性原理解决了重复数据块查询的瓶颈问题，利用有限的内存获得了较高的吞吐率；文献[25]针对一些缺少数据局部性特征的系统，提出基于文件相似性的特点来降低重复数据删除过程中查询次数的方法，以提高数据消冗性能。

重复数据删除技术在提高存储空间利用率的同时，会导致一个数据块被多个文件对象所引用，一个数据块的丢失会破坏文件的可用性，从而降低了数据存储可靠性。为了满足一些关键存储系统的可靠性需要，有研究者提出采用冗余复制或者纠删编码的方法提高存储可靠性：游（音译，L. You）在文献[7]通过量化分析表明引用度高的数据块应该获得较高的冗余度，并给出了采用冗余复制策略提高数据存储可靠性的方法，然而该文献没有给出数据块冗余度的定量计算方法，也没有对采用冗余机制后的数据可靠性、存储空间开销等因素进行定量的评估和分析；巴格瓦特（D. Bhagwat）在上述工作的基础上，给出了一种根据数据块引用度计算冗余度的方法[22]，并采用了冗余复制机制来提高数据存储可靠性，然而该工作没有对数据块冗余度、存储空间开销以及存储可靠性进行整体评估，没有给出最佳冗余度的计算方法；清华大学的刘川意在文献[23]提出了一种采用重复数据删除技术的归档存储系统的可靠性保证机制 R-ADMAD，该机制采用 ECC 编码机制对存储对象进行编码，并分布到一个冗余组的存储节点存放，并可通过一个分布动态恢复机制进行失效恢复。

2 重复数据删除技术的分类

2.1 基于重复内容识别方法的分类

(1)　基于散列识别

该方法通过数据的散列值来判断是否是重复数据。对于每个新数据块都生成一个散列，如果数据块的散列与存储设备上散列索引中的一个散列匹配，就表明该数据块是一个重复的数据块。Data Domain、飞康、昆腾的 DXi 系列设备都是采用 SHA-1、MD-5 等类似的散列算法来进行重复数据删除。

基于散列的方法存在内置的可扩展性问题。为了快速识别一个数据块是否已经被存储，这种基于散列的方法会在内存中拥有散列索引。随着数据块数量增加，该索引也随之增长。一旦索引增长超过了设备在内存中保存它所支持的容量，性能会急速下降，同时磁盘搜索会比内存搜索更慢。因此，目前大部分基于散列的系统都是独立的，可以保持存储数据所需的内存量与磁盘空间量的平衡。这样的设计使得散列表可以一直不致变得太大。

(2)　基于内容识别

该方法采用内嵌在数据中的文件系统的元数据识别文件，与其数据存储库中的其他版本进行逐字节地比较，找到该版本与前一个已存储版本的不同之处并为这些不同的数据创建一个增量文件。这种方法可以避免散列冲突，但是需要使用支持该功能的应用设备以提取元数据。

(3) 基于 ProtecTier 虚拟磁带库（VTL）的技术

这种方法像基于散列的方法产品那样将数据分成块，并且采用自有算法决定给定的数据块是否与其他数据块相似，然后与相似块中的数据进行逐字节的比较，以判断该数据块是否已经被存储。

2.2 基于去重粒度的分类

(1) 全文件层次的重复数据删除

以整个文件为单位来检测和删除重复数据，计算整个文件的哈希值，然后根据文件哈希值查找存储系统中是否存在相同的文件。这种方法的好处是在普通硬件条件下计算速度非常快；这种方法的缺点是即使不同文件存在很多相同的数据，也无法删除文件中的重复数据。

(2) 文件块消冗

将一个文件按不同的方式划分成数据块，以数据块为单位进行检测。该方法的优点是计算速度快、对数据变化较敏感。

(3) 字节级消冗

从字节层次查找和删除重复的内容，一般通过差异压缩策略生成差异部分内容。字节级消冗的优点是去重率比较高，缺点就是去重速度比较慢。

2.3 基于消冗执行次序的分类

(1) 在线式消冗

在线处理的重复数据删除是指在数据写入磁盘之前执行重复数据删除。其最大的优点是经济高效，可以降低对存储容量的需求，并且不需要保存还未进行重复数据删除的数据集。在线处理的重复数据删除减少了数据量，但同时也存在一个问题，处理本身会减慢数据吞吐速度。正是因为重复数据删除是在写入到磁盘之前进行的，因此重复数据删除处理本身就是一个单点瓶颈。

(2) 后处理式消冗

后处理的重复数据删除，也被称为离线重复数据删除，是在数据写到磁盘后再执行重复数据删除。数据先被写入到临时的磁盘空间，之后再开始重复数据删除，最后将经过重复数据删除的数据拷贝到末端磁盘。由于重复数据删除是数据写入磁盘后再在单独的存储设备上执行的，因此不会对正常业务处理造成影响。管理员可以随意制订重复数据删除的进程。通常先将备份数据保留在磁盘上再进行重复数据删除。企业在需要时可以更快速地访问最近存储的文件和数据。而后处理方式的最大问题在于它需要额外的磁盘空间来保存全部还未删除的重复数据集。

2.4 基于实现层次的分类

(1) 基于软件的重复数据删除

在软件层次，重复数据删除可以有两种集成方式，既可以将软件产品安装在专用的服务器上，也可以将其集成到备份/归档软件中。基于软件的重复数据删除的部署成本比较低，但是基于软件的重复数据删除在安装中容易中断运行，维护也比较困难。

基于软件的重复数据删除产品有 EMC 公司的 Avamar、Symantec 公司的 Veritas NetBackup 以及 Sepaton 公司的 DeltaStor 存储软件等。

(2)　基于硬件的重复数据删除

基于硬件的重复数据删除主要由存储系统自己完成数据的删减，例如：在虚拟磁带库系统、备份平台或者网络附加存储(NAS)等一般目的的存储系统中融入重复数据删除机制，由这些系统自身完成重复数据删除功能。

基于硬件的重复数据删除的优点是高性能、可扩展性和相对无中断部署，并且重复数据删除操作对上层的应用都是透明的。这种设备的缺点就是部署成本高于基于软件的重复数据删除。

目前基于硬件的重复数据删除系统主要包括虚拟磁带库（virtual tape library，VTL）和网络附加存储备份产品两大类，例如：Data Domain 公司的 DD410 系列、Diligent Technologies 公司的 ProtecTier VTL、昆腾公司的 DXi3500 和 DXi5500 系列、飞康的 VTL、ExaGrid Systems 公司的网络附加存储备份产品以及 NetApp 的 NearStore R200 和 FAS 存储系统。

3 相同数据重复数据删除技术

相同数据重复数据删除技术是将数据进行划分，找出相同的部分，并且以指针取代相同的数据存储。

3.1 相同文件重复数据删除技术

相同文件重复数据删除技术是以文件为粒度查找重复数据的方法[6]。如图 1 所示，以整个文件为单位计算出哈希值(SHA-1 或者 MD5)，然后与已存储的哈希值进行比较。如果发现相同的哈希值则认为该文件为重复文件，不进行存储；否则，该文件为新文件，将该文件及其哈希值存储到系统中。

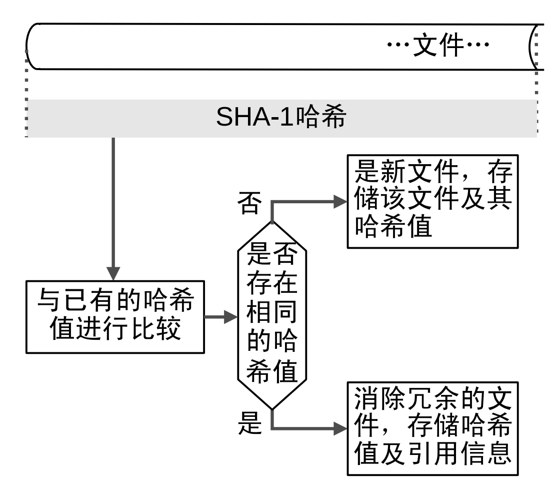


图1.　全文件消冗示意图

EMC 的 Centera 系统[1]、windows 的单实例存储系统[2]采用了这种数据消冗方法，利用 Windows2000 的 SIS（single instance storage）技术对具有 20 个不同 Windows NT 映像的服务器进行测试，结果表明总共节省了 58%的存储空间。该方法的优点是重复数据删除的速度比较快，缺点是不能删除不同文件内部的相同数据。

3.2 固定长度分块的重复数据删除技术

基于固定长度分块的重复数据删除方法如图 2 所示，将数据对象(文件)分成互不重叠的定长块，然后计算每个数据块的哈希值(SHA-1 值或者 MD5 值)，并将该哈希值与已存储的

哈希值进行检索比较，如果发现相同的哈希值，则认为该数据块是重复的数据块，不存储该数据块，只存储其哈希值及引用信息；否则， 该数据块是新数据块，存储该数据块、其哈希值及引用信息。

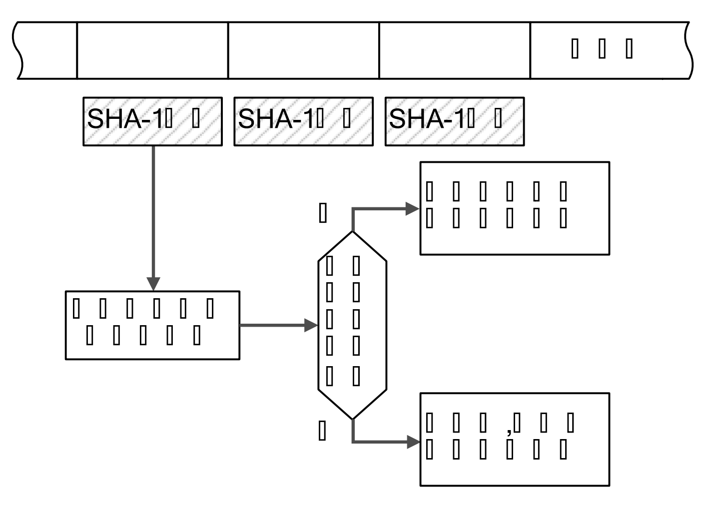


图2. 基于定长块的重复数据消除示意图

该方法存在的主要问题是：当向数据对象中插入数据或者从中删除数据时，会导致数据块边界无法对齐，严重影响重复数据删除的效果。如图 3 所示，数据对象的版本 1 生成了 n 个定长数据块 D_1、D_2、…、D_n，版本 2 在版本 1 的基础上插入了部分内容(阴影部分所示)，对版本 2 分块产生的数据块 D_1、D'_2、…、D'_n 中，只有 D_1 是重复的数据块，D'_2、…、D'_n 都不是重复的数据块，使得数据对象中从插入位置到结尾的重复数据都无法被消除，影响了消冗率。

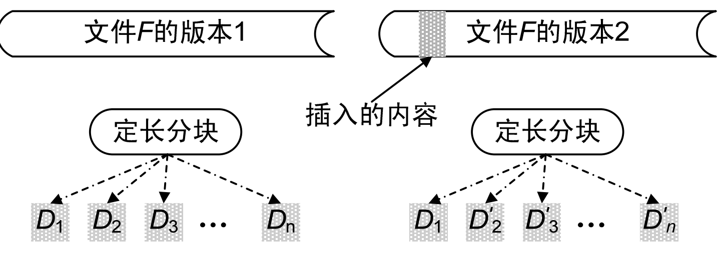


图3. 定长块重复数据消除技术所存在的问题

该方法已经在很多系统获得了应用，典型的应用是针对网络存储的 Venti 归档存储系统[3]，该系统采用该技术大约节省了 30%的空间。

3.3 基于内容分块算法的重复数据删除技术

针对上述问题，研究者提出了采用基于内容分块的重复数据删除方法[7](如图 4 所示)。该方法的思路是通过一个不断滑动的窗口来确定数据块分界点，采用拉宾指纹算法计算滑动窗口的指纹，如果满足预定条件，就将该窗口的开始位置作为数据块的结尾，这样通过不断滑动窗口并计算指纹实现对数据对象的分块。为了避免极端情况下，数据块过长或者过短的情况，可以设定数据块的下限和上限。对于每一个划分得到的数据块，就可以通过比较其哈希值来确定重复的数据块，具体过程与上面描述的相同。

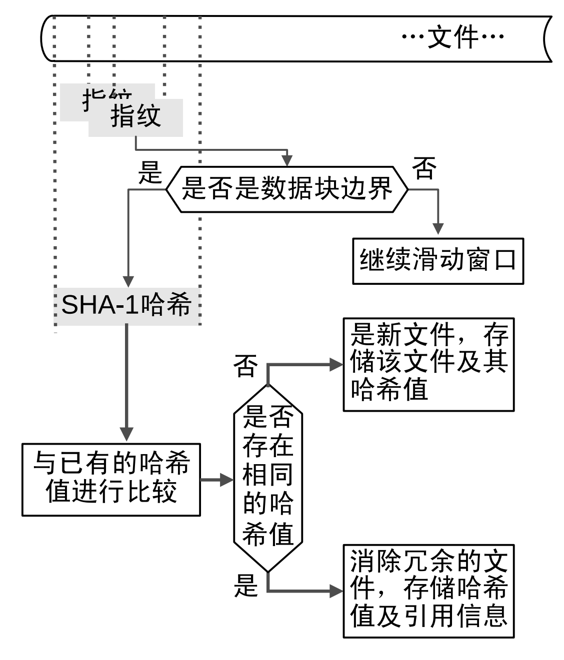


图4. 内容分块重复数据检测方法

因为数据块是基于内容而不是基于长度确定的，因此能够有效地解决固定长度分块的重复数据删除方法存在的问题。当数据对象中有内容插入或者删除时，如果插入或者删除的内容不在边界滑动窗口区域，该边界

不会改变。当插入的内容产生一个新的边界时，一个数据块会分成两个数据块，否则数据块不会变化。如果变化的内容发生在滑动窗口内，可能会破坏分界数据块，导致两个数据块合成一个数据块，或者两个数据块之间的边界发生变化，产生新的数据块。因此，插入或者删除内容只影响相邻的一个或者两个数据块，其余数据块不会受影响，这就使得该方法能够检测出对象之间更多的重复数据。如图 5 所示，当文件中插入部分内容后，分块时将该内容划分到一个数据块中，保持其后续的数据块不变，从而保证后面重复的数据块都能够被删除。

该方法的典型应用有 Shark[4]、Deep Store[5]等，并应用于低带宽网络文件系统（Low Bandwidth File System，LBFS）中。在低带宽网络文件系统中，系统对分块长度加上了上下边界长度，以避免数据块太长和太短的现象。

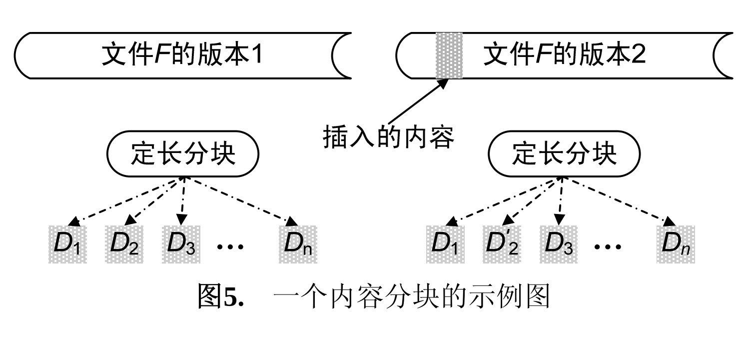


图5. 一个内容分块的示例图

3.4 基于滑动块的重复数据删除技术

内容划分块方法解决了字节插入和删除的问题，但又引入了变长块的存储问题。在存储系统中，变长块的存储组织比较复杂。针对该问题，出现了基于滑动块的重复数据删除检测消除方法[8]（如图 6 所示），解决了定长块和内容划分块所存在的问题。

滑动块方法采用了 *rsync Checksum*（文件同步备份校验和）和滑动窗口方法进行分块，*rsync Checksum* 算法具有计算速度快、效率高的优点。计算的校验和（*Checksum*）值与以前存储的值进行比较，如果匹配上，则计算数据块的 SHA-1 值进行比较来检测重复数据。

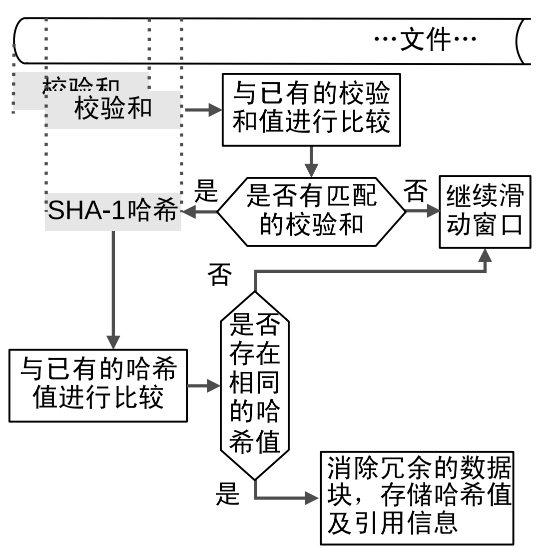


图6. 滑动块重复数据消除方法

如果发现重复数据块，则将重复数据块记录下来，并移动滑动窗口滑过该重复块，继续进行重复数据检测。另外，将从上个块结尾到新检测的重复块之间的数据块记录并存储下来。当 *Checksum* 值或者哈希值没有匹配上，则继续数据检测过程。如果在发现重复块之前滑动窗口移动的距离达到定长块的长度，则计算该块的哈希值，并将该值存储下来供将来进行数据块的校验。

滑动块方法通过检测对象的每一个块解决了数据插入问题。如果部分内容插入数据对象，只有周围的块发生变化，后面的块仍然能够通过该算法识别和检测。同理，当删除部分内容时，该部分之后的数据块不会受到影响，仍然可以采用该方法进行检测。

3.5 基于 fingerdiff 算法的重复数据删除技术

针对基于内容分块算法额外存储空间开销比较大的问题，研究者提出了 fingerdiff 算法[6]，其核心思想是将没有变化的块尽可能地合并，以减少数据块的元数据所占用的存储空

间。该技术包括三个主要过程：（1）一个文件按照基于内容分块算法进行数据块划分；（2）每个子块按照 fingerdiff 设置最大子块数进行合并；（3）每个块用哈希函数计算出它的指纹值，然后对比已存储的数据块指纹值，如果检测到相同的指纹值，则删除其对应的数据块，否则将大块进行拆分，找到最小的不同数据块进行存储，其余块仍然保持合并状态。

3.6 基于数据特征的重复数据消除算法

基于内容分块的块划分策略虽然在一定程度上解决了定长块所存在的问题，但是针对特定类型的数据文件，仍然无法获得较好的数据块划分。针对该问题，出现了基于数据特征的数据块划分策略。例如针对 PPT 类型文件的划分策略，根据 PPT 文件的格式按每页 PPT 划分成不同的数据块，从而有效地将相同的 PPT 页面消除。还有人提出了根据数据类型动态选择不同分块策略的重复数据删除技术，例如：针对 PPT 文件和 DOC 文件采用基于文件特征的重复数据消除策略，针对可执行文件采用定长块的分块策略。

4 相似数据重复数据删除技术

除了通过删除完全相同的数据可以实现数据消冗外，还可以通过相似数据的检测与编码节省存储空间，提高存储空间的利用率。相似数据重复数据删除包括相似数据检测和编码两个阶段，相似数据检测技术有以下几种。

Shingle 检测[1]技术通过为每个文档提取一组特征[9]将文档相似性问题简化为集合相似性问题。Shingle 检测技术简单易实现，适用范围广，但它的计算开销很高，而且检测相似数据的精度取决于 Shingle 的取样技术，容易出现较大的偏差。

布隆过滤器是一种用位数组表示的集合[10]，支持查询某个元素是否在该集合当中。布隆过滤器弥补了 shingle 检测技术计算开销大的缺陷，在性能和相似数据精度之间取得了平衡。布隆过滤器通过位操作进行数据匹配，所以速度快、计算开销很小。

通过模式匹配挖掘数据的特征也可以进行相似数据的检测。模式匹配技术的匹配算法是利用一定数量的公共字串来进行文件间的相似性查找与判别。该检测技术需要对整个文件进行扫描，所以开销也比较大。

在相似数据检测技术基础上，对有较大相似度的数据进行编码处理，同样能为整个系统节省大量的存储空间。然而相似数据压缩技术存在着编码效率和适用范围的问题。

5 重复数据删除的性能提升技术

重复数据删除技术在提高存储空间利用率的同时，对系统数据访问性能带来了一定的影响。这是因为重复数据的检测等过程要耗费大量的系统资源，严重影响了存储系统访问性能。针对该问题，目前也出现了一系列的解决方案。

针对内存空间无法容纳所有数据索引的问题，数据域（Data Domain）采用了三级查询[11]：布隆过滤器过滤、哈希缓冲查询和哈希文件查询。首先在内存中的布隆过滤器中进行查找，一个哈希用一个 $\boldsymbol{m}$ 位向量来概括在块索引中 $\boldsymbol{n}$ 个块指纹值的存在信息，如果布隆过滤器指出这个块不存在，则这个块一定不存在；如果布隆过滤器指出该数据块存在，表明该数据块可能存在，再到哈希缓存中进行查找；如果存在则说明该数据块存在，否则再到磁盘上去查

[1] 关于“Shingle”请参阅引文[9]

询。对于数据在磁盘上的组织采用了基于流的块排列技术，以有效利用数据的局部性，提高缓存的命中率。

针对数据访问局部性特征不明显的系统，研究者提出了基于文件相似性的特点来降低重复数据删除过程中的查询次数，以提高重复数据删除性能；另外，有人也采用了两阶段的重复数据删除机制[12]，通过将随机的小磁盘读写调整为序列化的大的磁盘读写提高重复数据删除的吞吐率；还有人采用了两层次的索引技术来降低磁盘读写次数[13]，提高重复数据删除的吞吐率。

分析现有技术可以看出，提高重复数据删除吞吐率的关键是降低磁盘读写次数，现有方法都是通过各种策略来尽量减少数据块检索过程中磁盘的读写次数。

6 重复数据删除技术的应用

6.1 数据备份系统

重复数据删除技术为数据保护领域带来革命性突破，有效地改善了基于磁盘数据保护的成本效益。因为在传统数据保护中无法实现重复数据删除，往往采用廉价的磁带库作为备份设备。磁带备份在备份窗口、恢复速度方面难以满足用户的需求。现在，基于磁盘的数据保护方案如虚拟磁带库被广泛采用，并且在未来会继续增长。备份到虚拟磁带库或其他基于磁盘的备份已经缩小了备份窗口，改善了备份和恢复能力，但由于数据量的不断增加，我们所要备份的数据越来越多，面临容量膨胀的压力。重复数据删除技术的出现为最大限度降低存储容量找到有效的方法。

6.2 归档存储系统

重复数据删除技术对归档存储也非常重要。由于参考数据的数量不断增长，而“法规遵从”要求数据在线保留的时间更长，并且由于高性能需求需要采用磁盘进行归档，因此，企业一旦真正开始进行数据的归档存储就会面临成本问题。理想的归档存储系统应能满足长期保存归档数据的需求，并且其总体拥有成本也必须低于生产系统的成本。重复数据删除技术通过消除冗余实现高效率的归档存储，从而实现最低的成本。 目前，归档存储系统的重复数据删除技术主要是基于哈希方法，产品的销售形态是以内容寻址存储（CAS，Content Addressable Storage)技术为主，分为纯软件和存储系统两类。

6.3 远程灾备系统

在远程灾备系统中，需要将大量的数据迁移到异地系统。随着数据量的不断增长，数据传输的压力越来越大，通过重复数据删除技术在数据传输前检测并删除重复的数据，可以有效地减少传输的数据量，提高数据传输速度，典型产品如飞康的 MicroScan 软件。

7 总结和发展趋势

通过上述分析可以看出，根据应用的实际需求，研究者分别对消冗率、性能以及可靠性展开了大量的研究，取得了很多有价值的研究成果。但在面向服务的存储系统中，不同用户或者应用对于重复数据删除率、性能及可靠性等属性具有不同的需求，为了满足这种多样化需求，就需要根据用户需求对重复数据删除率、性能及可靠性等属性进行动态调整[21]。然而目前这方面的研究还比较初步，无法满足面向服务的大规模存储系统的应用需求。

另外，存储规模的不断扩大导致能耗问题越来越突出，目前存储系统能耗已经达到 IT 系统能耗的 40%，并且这个比例还在不断增加。虽然研究者针对存储系统的能耗问题以及能耗优化方法进行了一系列的研究工作，取得了很多有价值的研究成果，然而忽略了重复数据删除对存储系统能耗的影响。根据前期的研究结果发现，重复数据删除机制会增加存储系统的能耗，并且会影响现有能量优化机制的效果，服务质量和能耗之间存在一定的矛盾关系。因此，在对重复数据删除的服务质量进行研究的过程中，还需要考虑系统的能耗因素，实现对服务质量和能量开销的有效平衡和调整。

综上所述，我们可以看到以下需要研究的问题：（1）如何挖掘不同类型的数据特征，快速准确地检测到重复数据，同时有效降低空间开销；(2) 如何克服数据相似性检测技术设计上存在的局限性，在融合各技术特征的同时，通过结合统计学和数据挖掘领域的各种技术，对数据特征进行充分的分析和挖掘，通过提高对其规律性的认识来弥补重复数据删除技术上的不足，提高整体系统的性能；(3) 如何根据不断出现的新的压缩理论与技术或更有效的数学模型，通过引进压缩算法开发新的技术或将已有技术结合在一起，有效地优化储存空间；(4) 如何在已有的基于增加冗余数据，具有简单高效特点的可靠性技术的基础上，克服其存储开销和系统性能方面存在的局限性，针对不同的数据类型，适度地增加冗余数据来提高系统的可靠性或通过引入其他机制改进可靠性设计；(5) 如何在应用重复数据删除技术的过程中，在简单性和性能两方面做出这种选择，在融合各种现有技术的同时，提供通用型、可扩展性和自适应性，尽可能减少重复数据检测和删除所带来的系统开销。

参考文献：

[1] H. M. Sung,W. Y. Lee, J. Kim, and Y.W. Ko, Design and Implementation of Clustering File Backup Server Using File Fingerprint, *Soft. Eng., Arti. Intel., Net. & Para./Distri. Comp.*, 2008, pp.61-73.

[2] C. Liu, D. Ju, Y. Gu, Y. Zhang and D. Wang,Semantic Data De-duplication for Archival Storage Systems, *the 13th Asia-Pacific conference on Computer Systems Architecture Conference*, Aug. 2008, pp.1-9.

[3] Y. Won, R. Kim, J. Ban, and J. Hur, PRUN: Eliminating Information Redundancy for Large Scale Data Backup System, *Proceedings of the 2008 International Conference on Computational Sciences and Its Applications*, 2008, pp 139-144.

[4] B. ZHU, H. LI, AND H. PATTERSON, Avoiding the disk bottleneck in the data domain deduplication file system. In *Proceedings of the 6th USENIX Conference on File And Storage Technologies* (*FAST'08*), San Jose, California,February 2008, pp. 1-14.

[5] S. Quinlan，S. Dorward. Venti: A new approach to archival storage. In *Proceedings of the 2002 Conference on File and Storage Technologies* (*FAST*), Monterey, California, USA, 2002, pp.89-101.

[6] L. L. You. Efficient Archival Data Storage. *Techncial Report UCSC-SSRC-06-04*, University of California, Santa Cruz, June 2006.

[7] L You, K Pollack and D. Long, Deep store: An archival storage system architecture, *Proceedings of the 21st IEEE International Conference on Data Engineering*, IEEE Press, Santa Cruz, CA, USA,2005, pp. 804–815.

[8] S. Annapureddy, M. J. Freedman, and D. Mazières. Shark: Scaling file servers via cooperative caching. In *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation (NSDI'05)*, Boston, MA, May 2005, pp. 129-142.

[9] A. Z. Broder. Identifying and filtering near-duplicate documents. In: Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching. Montreal, Canada: Springer-Verlag New York, Inc., Jun. 2000. 1-10.

[10] M. W. Storer, K. M. Greenan, D. D. E. Long, and E. L. Miller. Secure data deduplication. In *Proceedings of the 2008 ACM Workshop on Storage Security and Survivability*, Oct. 2008, pp 1-10.

[11] G. Forman, K. Eshghi, J. Suermondt， Efficient Detection of Large Scale Redundancy in Enterprise File Systems, *HPL-2008-30R2*, HP Laboratories, 2008.

[12] D. Trendafilov, N. Memon, and T. Suel. zdelta: An efficient delta compression tool. *Technical Report TR-CIS-2002-02*, Polytechnic University, June 2002.

[13] H. S. Gunawi, N. Agrawal, A. C. Arpaci-Dusseau, R. H. Arpaci-Dusseau, and J. Schindler. Deconstructing commodity storage clusters. In *Proceedings of the 32nd Int'l Symposium on Computer Architecture*, June 2005, pp. 60–71.

[14] W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur.. Single instance storage in Windows 2000. *http://research.microsoft.com/farsite/WSS2000.pdf*.

[15] M. O. Rabin. Fingerprinting by random polynomials. *Technical Report TR-15-81*, Center for Research in Computing Technology, Harvard University, 1981.

[16] C. Liu, Y. Lu, C. Shi, and G. Lu, ADMAD: Application-Driven Metadata Aware De-duplication Archival Storage System, *the 15th IEEE international workshop on Storage Network Architecture and Parallel I/O(SNAPI '08)*, 2008, pp.29-35.

[17] F. Douglis, A. Iyengar. Application-specific deltaencoding via resemblance detection. In *Proceedings of the 2003 USENIX Annual Technical Conference*, San Antonio, Texas, June 2003.

[18] L. Xu, Hydra: A Platform for Survivable and Secure Data Storage Systems, Proc. *Int'l Workshop Storage Security and Survivability*, Virginia, USA, Nov. 2005.

[19] J.J. Wylie, M.W. Bigrigg, J.D. Strunk, and G.R. Ganger. Survivable Information Storage Systems, *IEEE Computer*, 33(8),2000: 61-68, Aug 2000.

[20] T. Yang, H. Jiang, D. Feng and Z. Niu. DEBAR: A Scalable High-Performance De-duplication Storage System for Backup and Archiving, *Technical Report TR-UNL-CSE-2009-0004* , HUST ,2008

[21] 舒继武, 网络存储领域若干技术发展与启示, http://www.ccf.org.cn/web/resource/shujiwu.pdf, 2008

[22] D. Bhagwat, K. Pollack, D. D. E. Long, and T. Schwarz, Providing High Reliability in a Minimum Redundancy Archival Storage System, Pr*oceedings of the 4th ACM international workshop on Storage security and survivability*, Virginia, USA, 2008, pp. 1-10.

[23] Chuanyi Liu, Yu Gu, Linchun Sun, Bin Yan, Dongsheng Wang: R-ADMAD: high reliability provision for large-scale de-duplication archival storage systems. *ICS 2009*: 370-379.

[24] Mark Lillibridge, Kave Eshghi, Deepavali Bhagwat, Vinay Deolalikar, Greg Trezise, and Peter Camble. Sparse indexing: large scale, inline deduplication using sampling and locality. In *FAST '09: Proccedings of the 7th conference on File and storage technologies*, page 111-123, Berkeley, CA, USA, 2009. USENIX Association.

[25] Deepavali Bhagwat, Kave Eshghi, Darrell D.E. Long, Mark Lillibridge. Extreme Binning: Scalable, Parallel Deduplication for Chunk-based File Backup. In *IEEE MASCOTS 2009*, London, UK, September, 21st, 2009

作者简介：

王树鹏： 中国科学院计算技术研究所信息安全研究中心数据存储与保护小组组长，博士 wangshupeng@software.ict.ac.cn

云晓春： 中国科学院计算技术研究所研究员

郭 莉： 中国科学院计算技术研究所信息安全研究中心主任，研究员